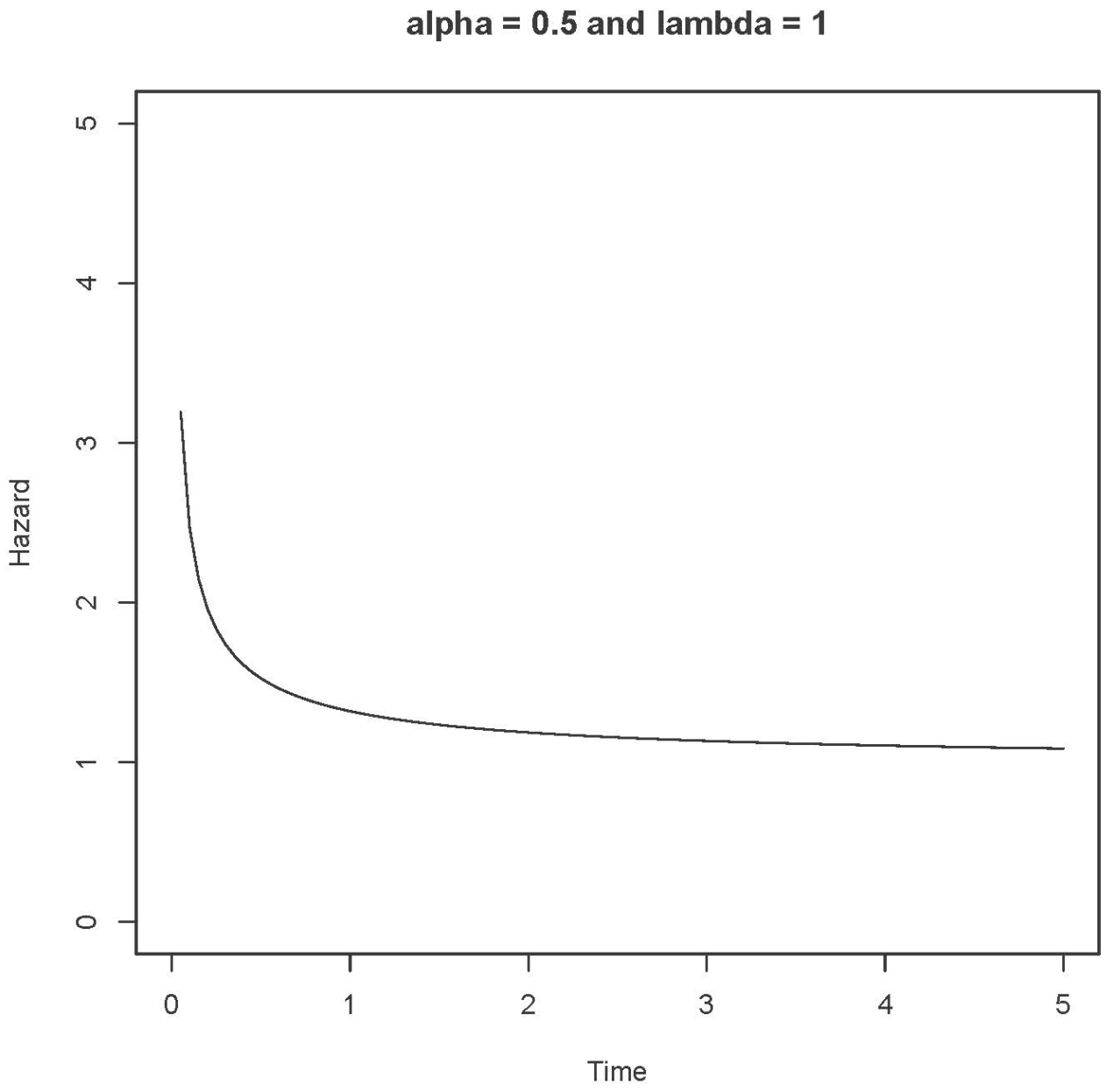
Exploring the hazard function of the gamma distribution*

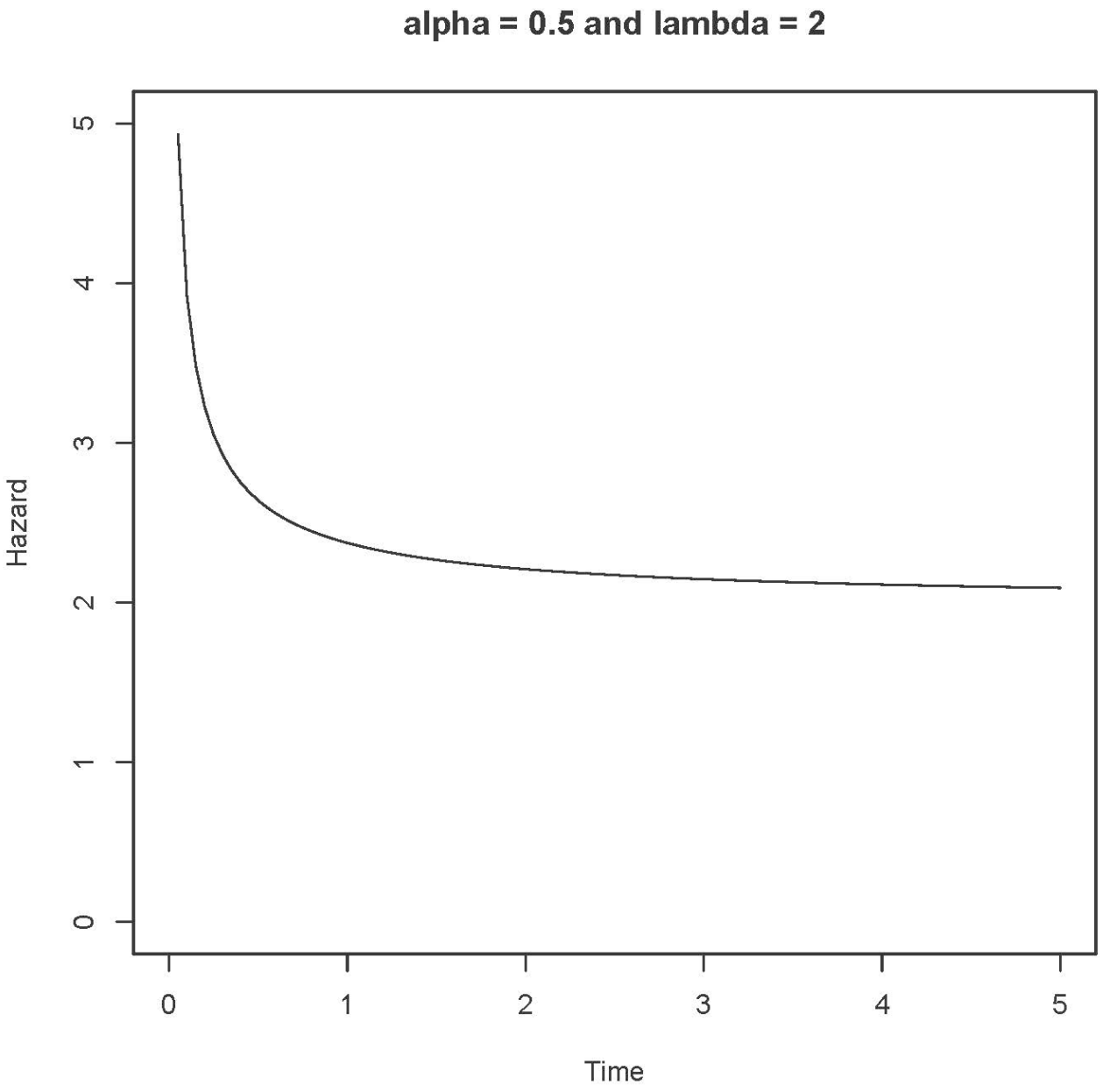
```
> # Hazard function of Gamma(alpha,lambda) distribution: : h(t) = f(t)/S(t)
>
> # You might think the following would work, but watch.
> # Remember that a gamma distribution with alpha=1 is exponential.
> # The hazard function of an exponential distribution is a constant lambda.
>
> Time = 1:10
>
> alpha=1; lambda=1
> dgamma(Time,shape=alpha,rate=lambda) / (1-pgamma(Time,shape=alpha,rate=lambda))
 [1] 1 1 1 1 1 1 1 1 1 1
>
> alpha=1; lambda=2
> dgamma(Time,shape=alpha,rate=lambda) / (1-pgamma(Time,shape=alpha,rate=lambda))
 [1] 2 2 2 2 2 2 2 2 2 2
>
> alpha=1; lambda=3
> dgamma(Time,shape=alpha,rate=lambda) / (1-pgamma(Time,shape=alpha,rate=lambda))
 [1] 3.000000 3.000000 3.000000 3.000000 3.000000 3.000000 3.000000 2.999999
 [9] 3.000051 2.999501
>
> alpha=1; lambda=4
> dgamma(Time,shape=alpha,rate=lambda) / (1-pgamma(Time,shape=alpha,rate=lambda))
 [1] 4.000000 4.000000 4.000000 4.000000 4.000000 3.999999 3.999960 4.002409
 [9] 4.178481      Inf
>
> alpha=1; lambda=10
> dgamma(Time,shape=alpha,rate=lambda) / (1-pgamma(Time,shape=alpha,rate=lambda))
 [1] 10.000000 10.000000  9.998336       Inf       Inf       Inf       Inf
 [8]       Inf       Inf       Inf
>
> # Use logs to get around division by a number close to zero
> alpha=1; lambda=10
> logh = dgamma(Time,shape=alpha,rate=lambda,log=TRUE) -
+     pgamma(Time,shape=alpha,rate=lambda, lower.tail=FALSE,log.p=TRUE)
> exp(logh)
 [1] 10 10 10 10 10 10 10 10 10 10
```

* This document is free and open source. See last page for copyright information.

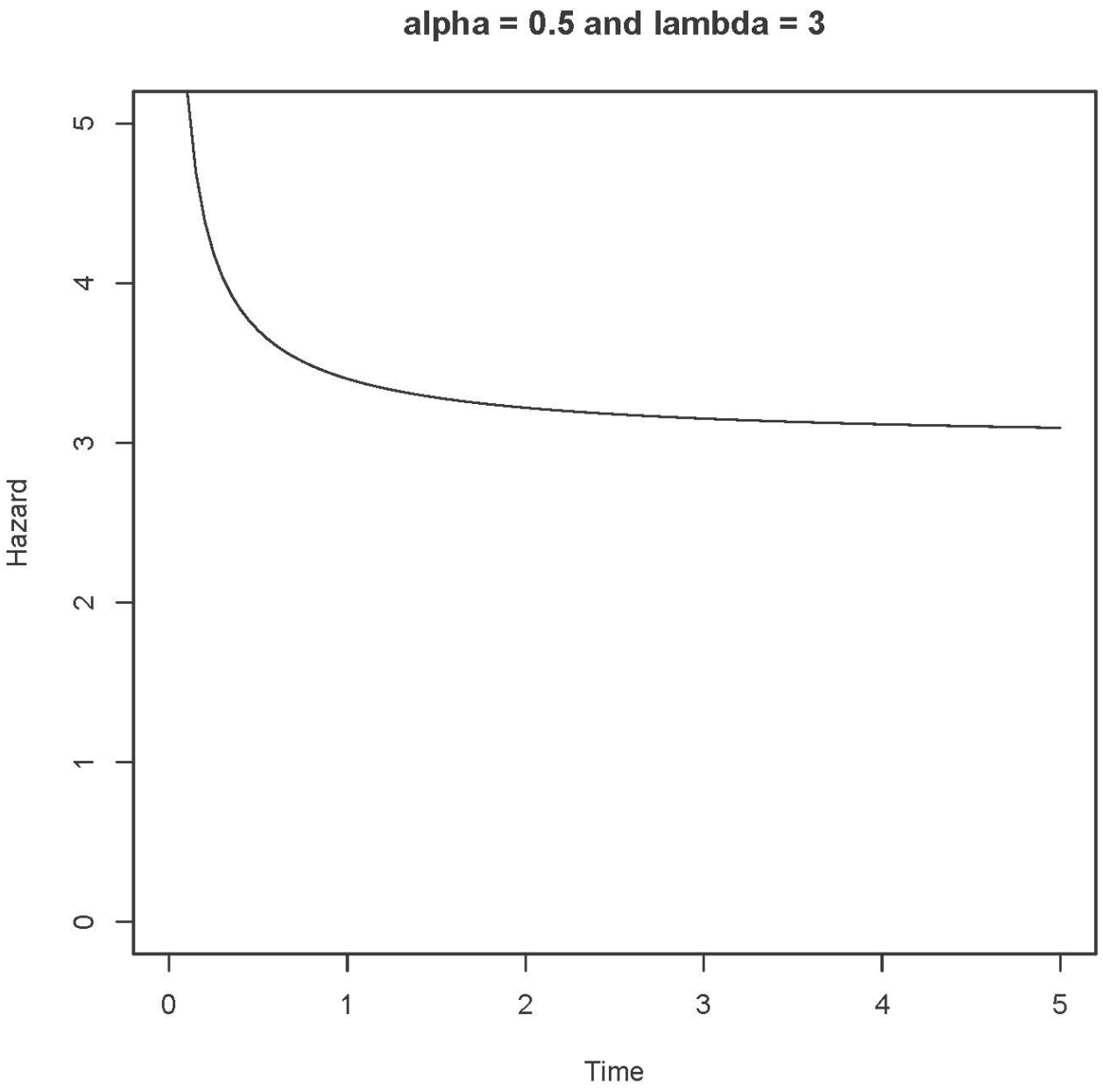
```
> # Now plot
> alpha=1/2;lambda=1
> Time = seq(from=0,to=5,length=101)
> logh = dgamma(Time,shape=alpha,rate=lambda,log=TRUE) -
+     pgamma(Time,shape=alpha,rate=lambda, lower.tail=FALSE,log.p=TRUE)
> Hazard = exp(logh)
> tstring = paste('alpha =',alpha,'and lambda =',lambda)
> plot(Time,Hazard,type='l',main=tstring,ylim=c(0,5)) # That's a lower case ell
> # Notice explicit limits on y to keep all plots on the same scale.
```



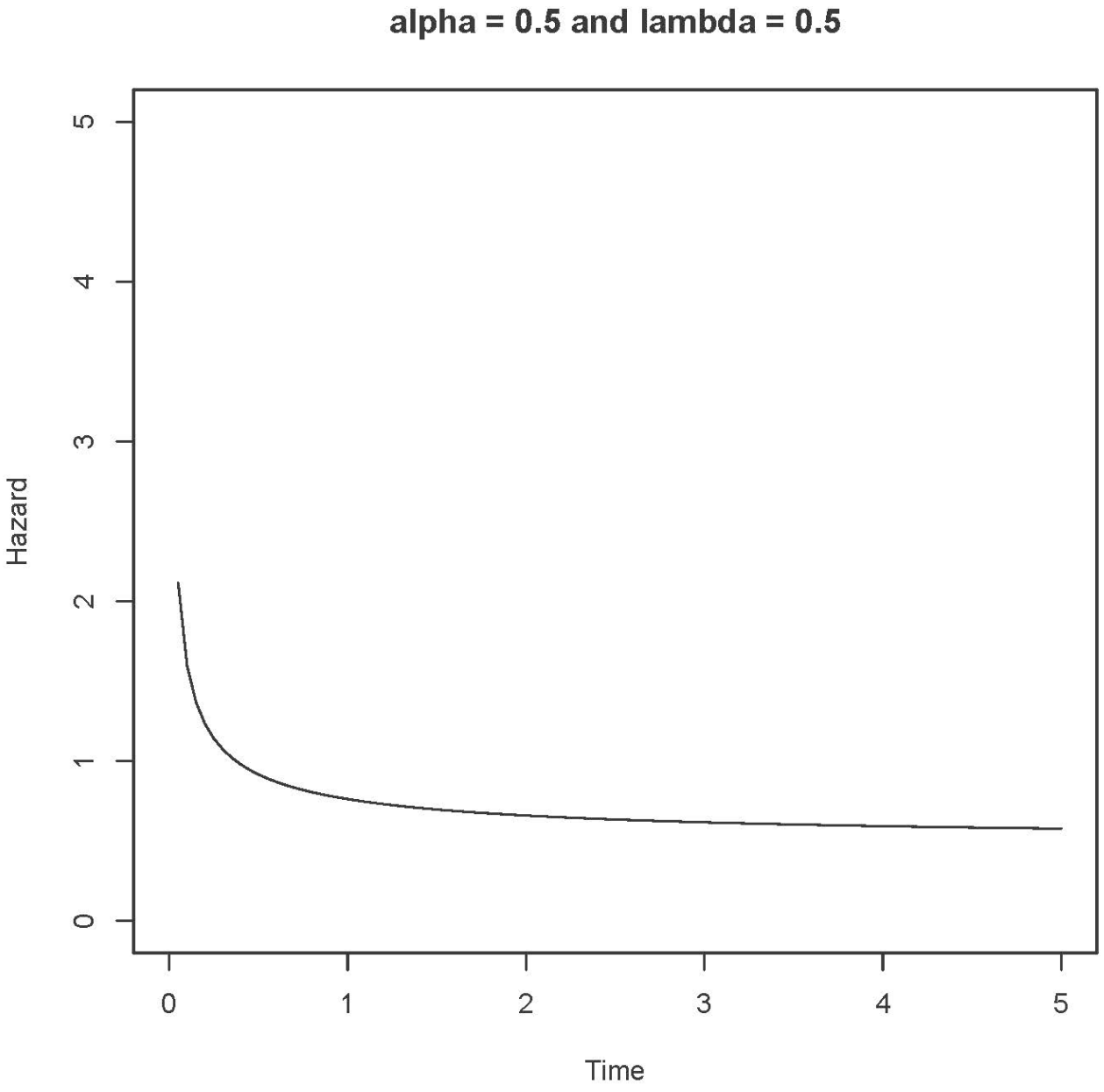
```
> alpha=1/2;lambda=2
> Time = seq(from=0,to=5,length=101)
> logh = dgamma(Time,shape=alpha,rate=lambda,log=TRUE) -
+     pgamma(Time,shape=alpha,rate=lambda, lower.tail=FALSE,log.p=TRUE)
> Hazard = exp(logh)
> tstring = paste('alpha =',alpha,'and lambda =',lambda)
> plot(Time,Hazard,type='l',main=tstring,ylim=c(0,5)) # That's a lower case ell
```



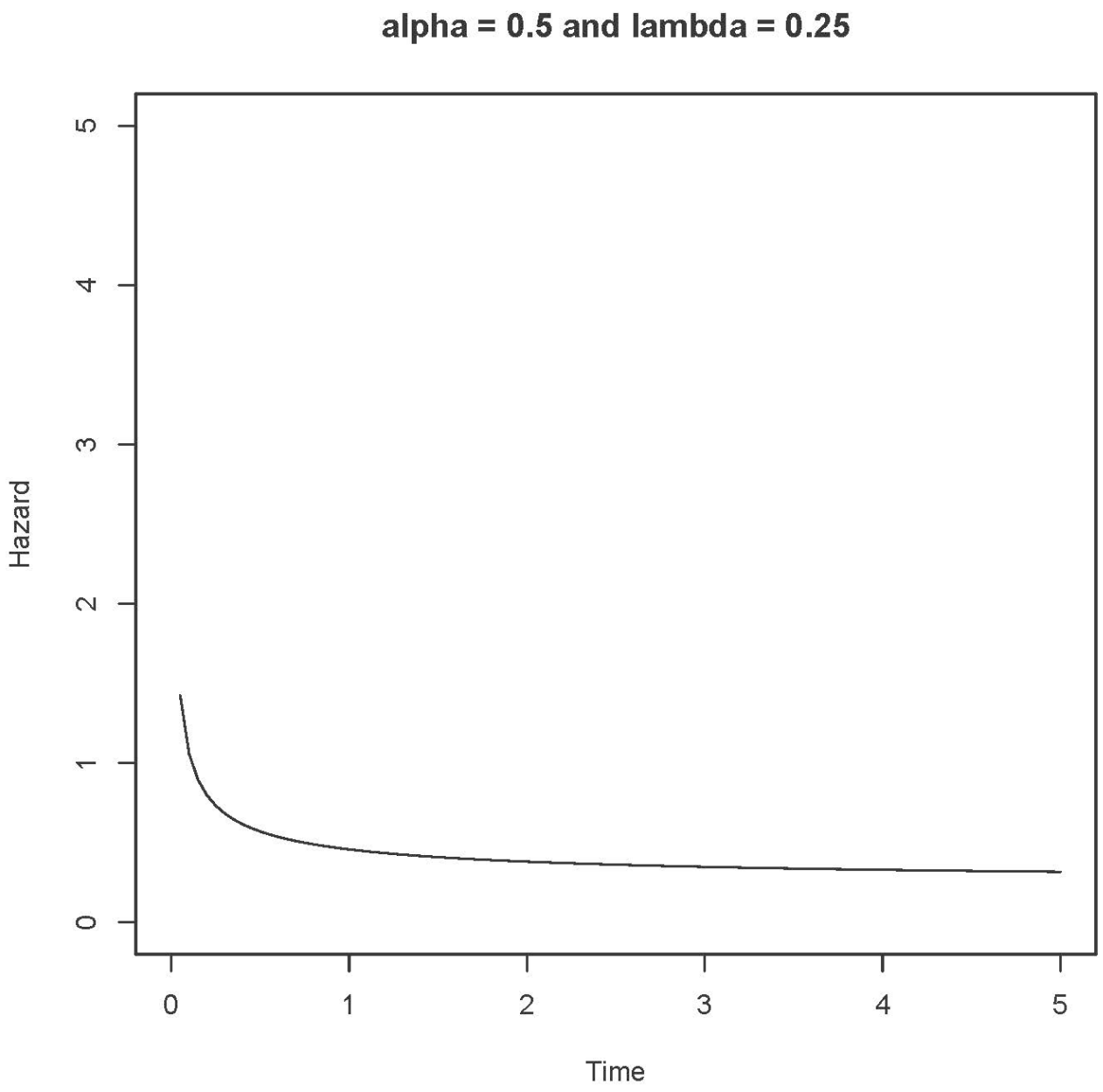
```
> alpha=1/2;lambda=3
> Time = seq(from=0,to=5,length=101)
> logh = dgamma(Time,shape=alpha,rate=lambda,log=TRUE) -
+     pgamma(Time,shape=alpha,rate=lambda, lower.tail=FALSE,log.p=TRUE)
> Hazard = exp(logh)
> tstring = paste('alpha =',alpha,'and lambda =',lambda)
> plot(Time,Hazard,type='l',main=tstring,ylim=c(0,5)) # That's a lower case ell
```



```
> # Try lambda small
> alpha=1/2;lambda=1/2
> Time = seq(from=0,to=5,length=101)
> logh = dgamma(Time,shape=alpha,rate=lambda,log=TRUE) -
+     pgamma(Time,shape=alpha,rate=lambda, lower.tail=FALSE,log.p=TRUE)
> Hazard = exp(logh)
> tstring = paste('alpha =',alpha,'and lambda =',lambda)
> plot(Time,Hazard,type='l',main=tstring,ylim=c(0,5)) # That's a lower case ell
```

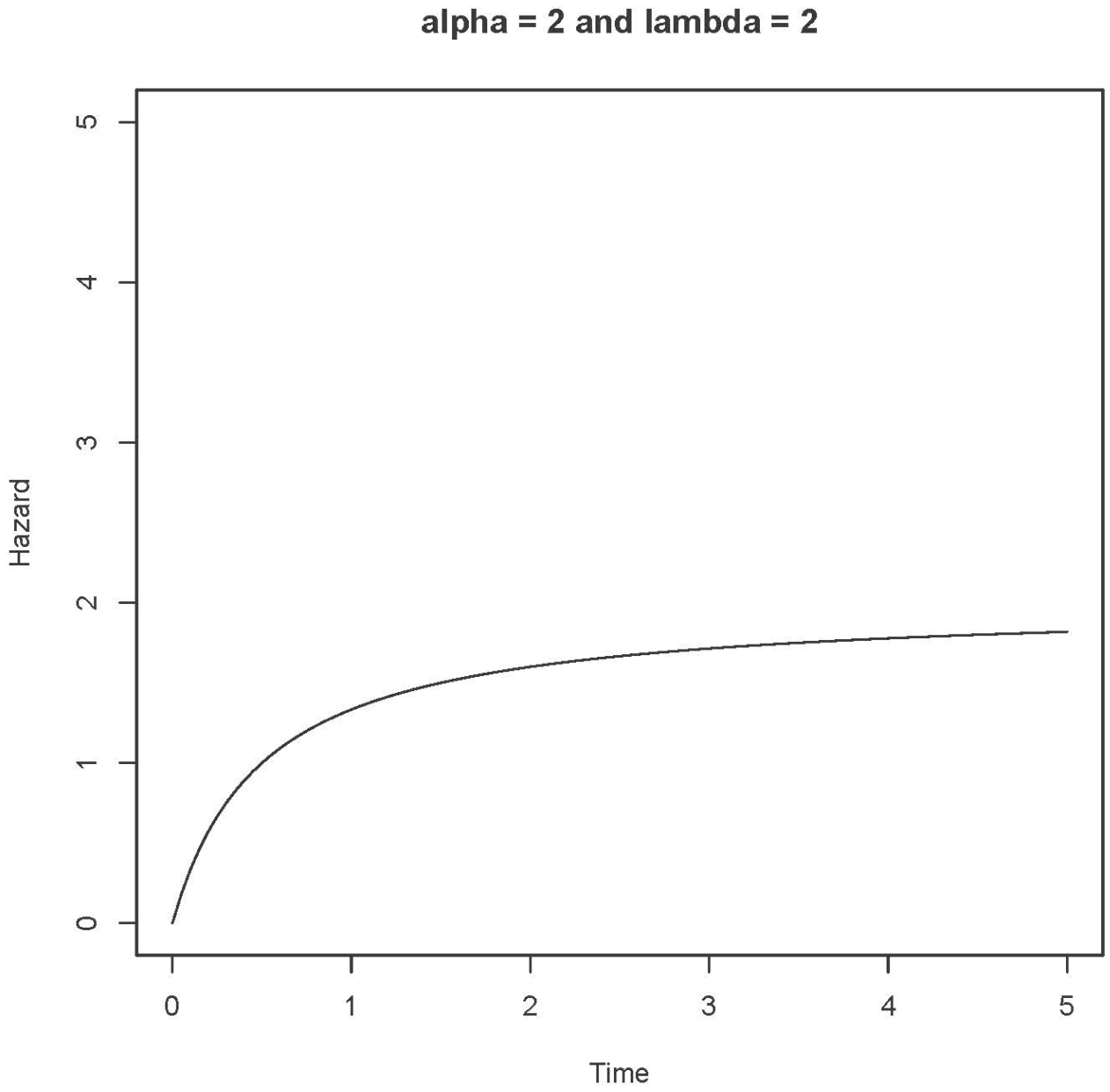


```
> alpha=1/2;lambda=1/4
> Time = seq(from=0,to=5,length=101)
> logh = dgamma(Time,shape=alpha,rate=lambda,log=TRUE) -
+     pgamma(Time,shape=alpha,rate=lambda, lower.tail=FALSE,log.p=TRUE)
> Hazard = exp(logh)
> tstring = paste('alpha =',alpha,'and lambda =',lambda)
> plot(Time,Hazard,type='l',main=tstring,ylim=c(0,5)) # That's a lower case ell
```

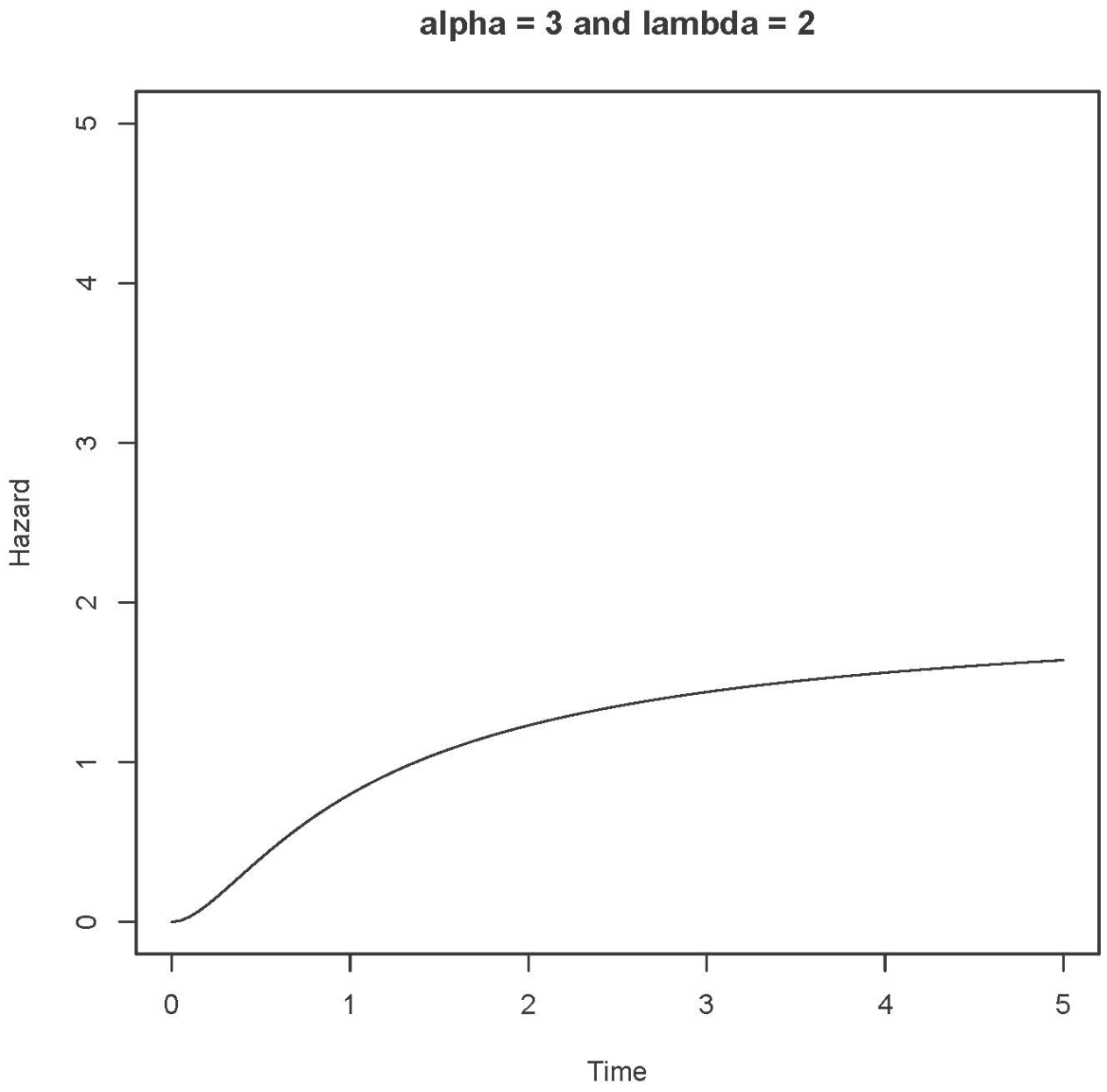



```
>
> # Lambda appears to control the overall level of the function, with larger
> # numbers higher. Investigate alpha, initially holding lambda fixed at 2.
```

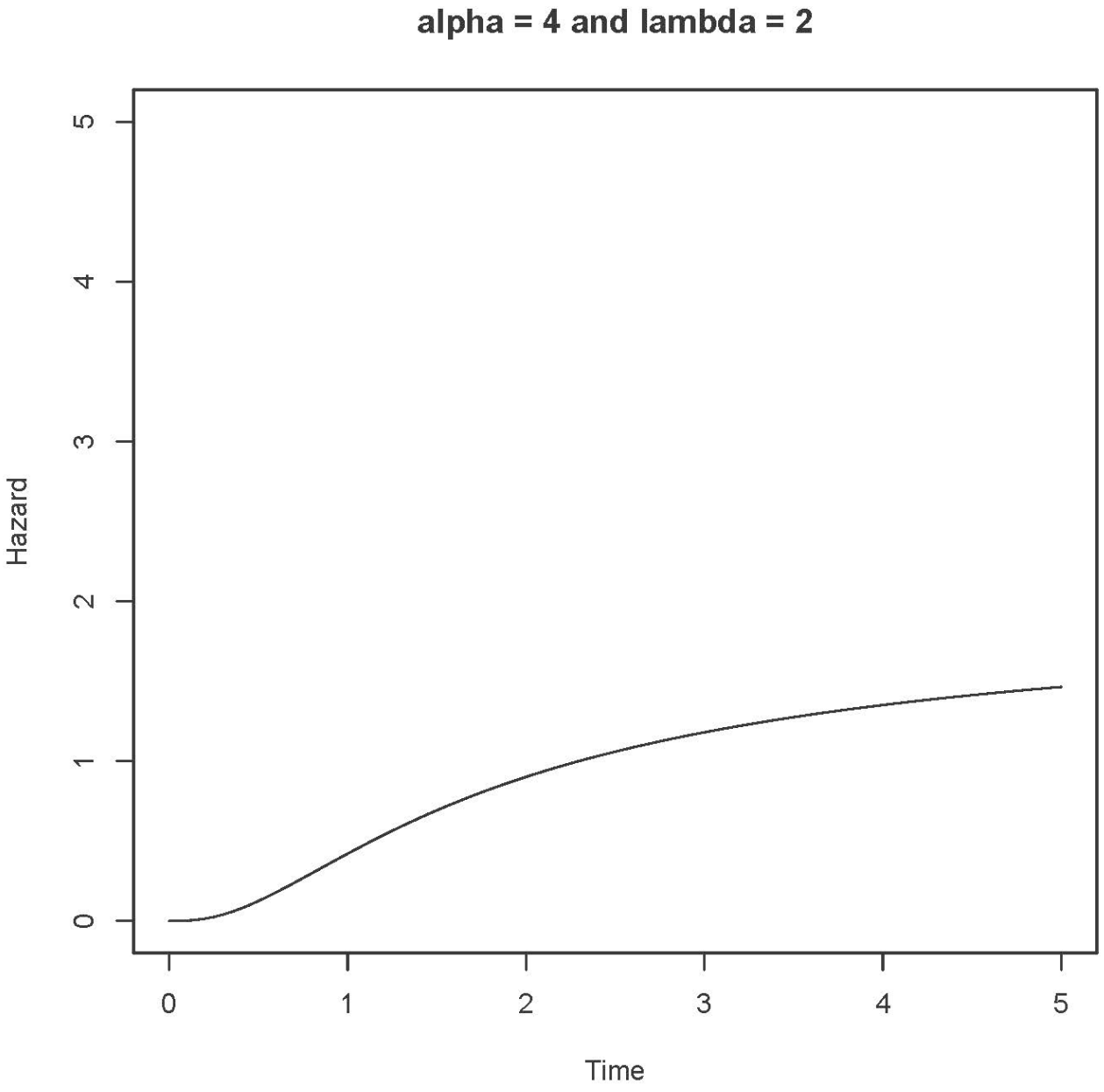
```
> # Investigate alpha, initially holding lambda fixed at 2
>
> alpha=2;lambda=2
> Time = seq(from=0,to=5,length=101)
> logh = dgamma(Time,shape=alpha,rate=lambda,log=TRUE) -
+     pgamma(Time,shape=alpha,rate=lambda, lower.tail=FALSE,log.p=TRUE)
> Hazard = exp(logh)
> tstring = paste('alpha =',alpha,'and lambda =',lambda)
> plot(Time,Hazard,type='l',main=tstring,ylim=c(0,5)) # That's a lower case ell
```



```
> alpha=3;lambda=2
> Time = seq(from=0,to=5,length=101)
> logh = dgamma(Time,shape=alpha,rate=lambda,log=TRUE) -
+     pgamma(Time,shape=alpha,rate=lambda, lower.tail=FALSE,log.p=TRUE)
> Hazard = exp(logh)
> tstring = paste('alpha =',alpha,'and lambda =',lambda)
> plot(Time,Hazard,type='l',main=tstring,ylim=c(0,5)) # That's a lower case ell
```

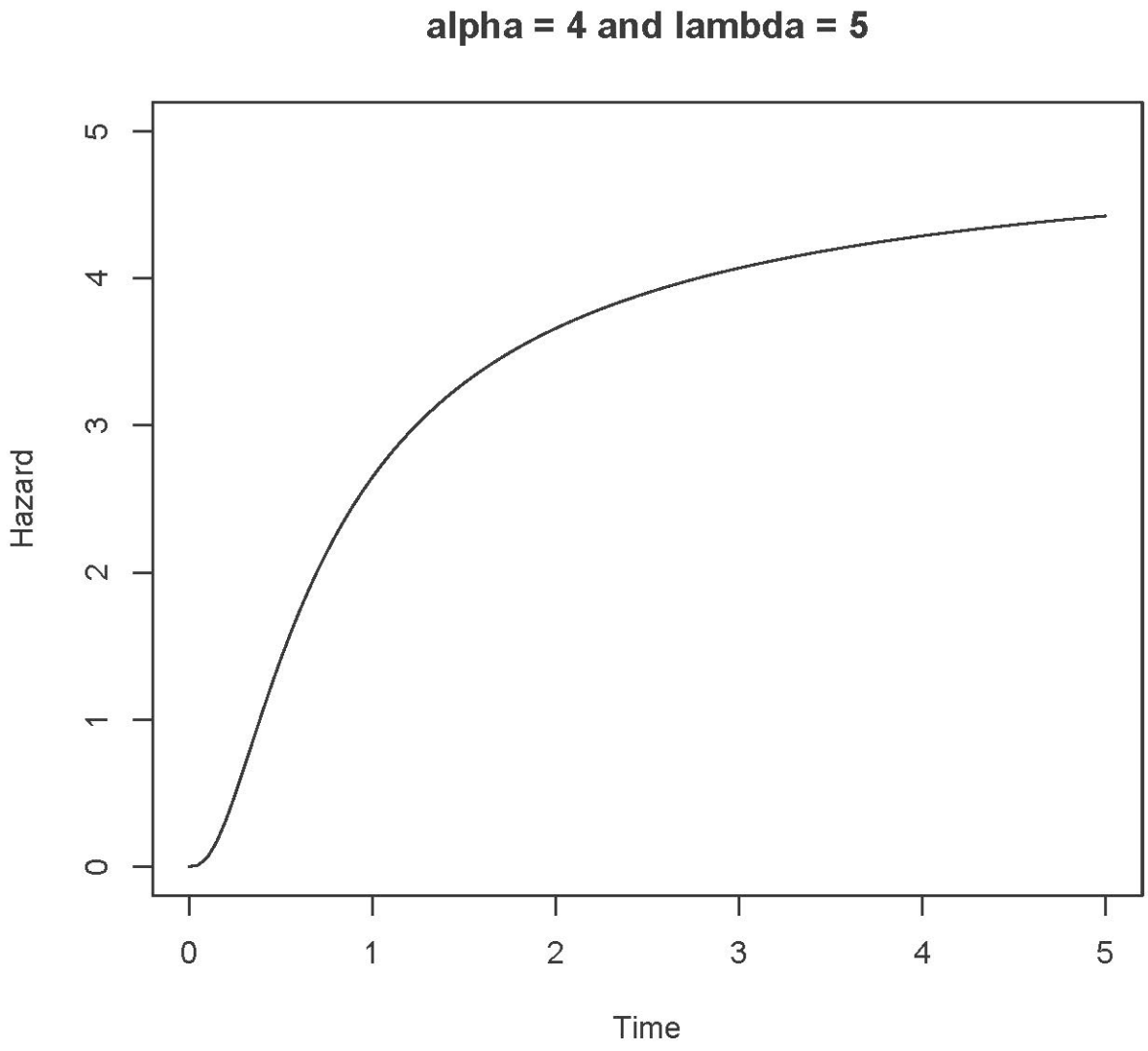


```
> alpha=4;lambda=2
> Time = seq(from=0,to=5,length=101)
> logh = dgamma(Time,shape=alpha,rate=lambda,log=TRUE) -
+     pgamma(Time,shape=alpha,rate=lambda, lower.tail=FALSE,log.p=TRUE)
> Hazard = exp(logh)
> tstring = paste('alpha =',alpha,'and lambda =',lambda)
> plot(Time,Hazard,type='l',main=tstring,ylim=c(0,5)) # That's a lower case ell
>
```

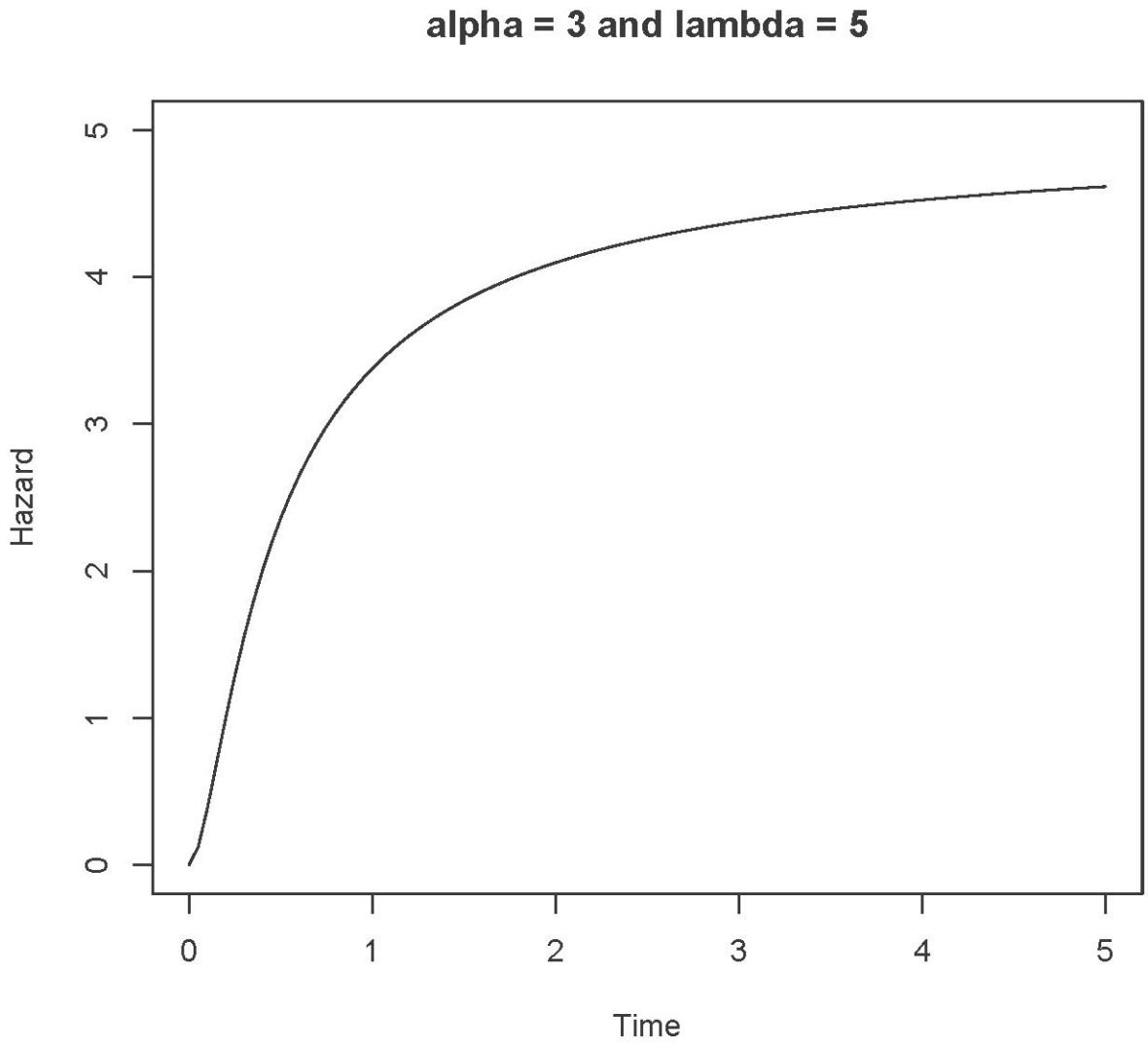



```
> # Increase lambda to 5
```

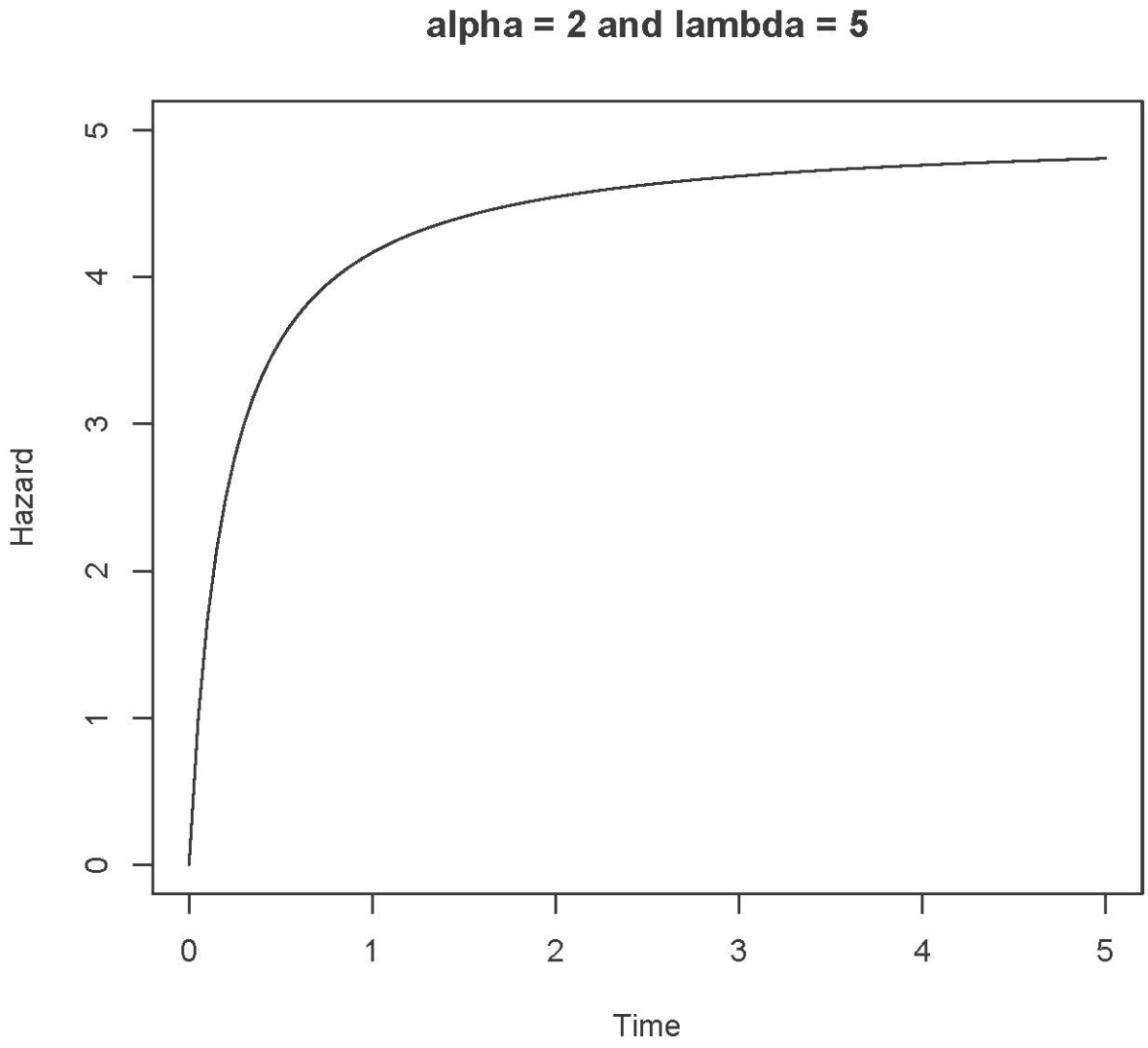
```
> # Increase lambda to 5
>
> alpha=4;lambda=5
> Time = seq(from=0,to=5,length=101)
> logh = dgamma(Time,shape=alpha,rate=lambda,log=TRUE) -
+     pgamma(Time,shape=alpha,rate=lambda, lower.tail=FALSE,log.p=TRUE)
> Hazard = exp(logh)
> tstring = paste('alpha =',alpha,'and lambda =',lambda)
> plot(Time,Hazard,type='l',main=tstring,ylim=c(0,5)) # That's a lower case ell
```



```
> alpha=3;lambda=5
> Time = seq(from=0,to=5,length=101)
> logh = dgamma(Time,shape=alpha,rate=lambda,log=TRUE) -
+     pgamma(Time,shape=alpha,rate=lambda, lower.tail=FALSE,log.p=TRUE)
> Hazard = exp(logh)
> tstring = paste('alpha =',alpha,'and lambda =',lambda)
> plot(Time,Hazard,type='l',main=tstring,ylim=c(0,5)) # That's a lower case ell
```

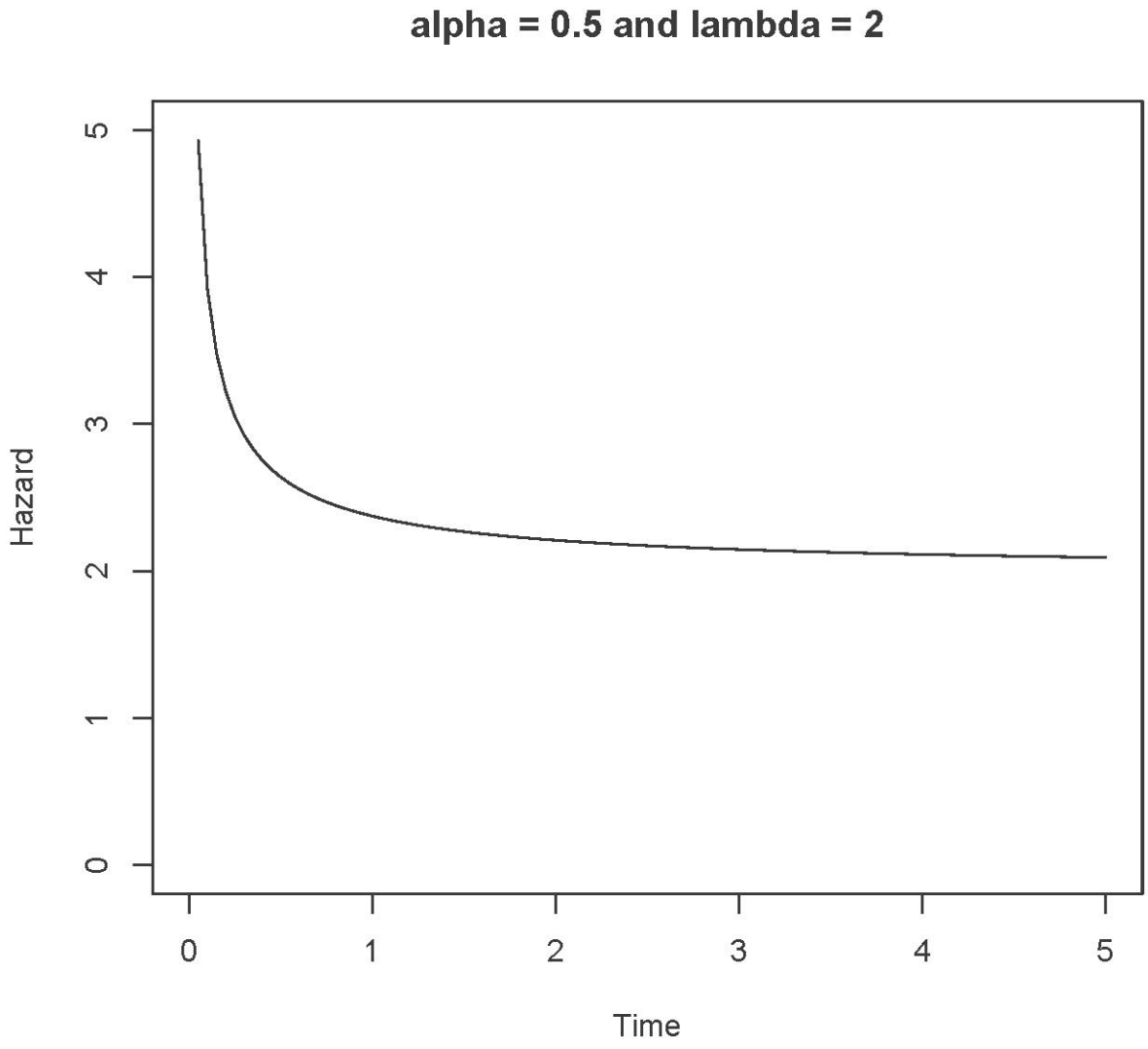


```
> alpha=2;lambda=5
> Time = seq(from=0,to=5,length=101)
> logh = dgamma(Time,shape=alpha,rate=lambda,log=TRUE) -
+     pgamma(Time,shape=alpha,rate=lambda, lower.tail=FALSE,log.p=TRUE)
> Hazard = exp(logh)
> tstring = paste('alpha =',alpha,'and lambda =',lambda)
> plot(Time,Hazard,type='l',main=tstring,ylim=c(0,5)) # That's a lower case ell
```

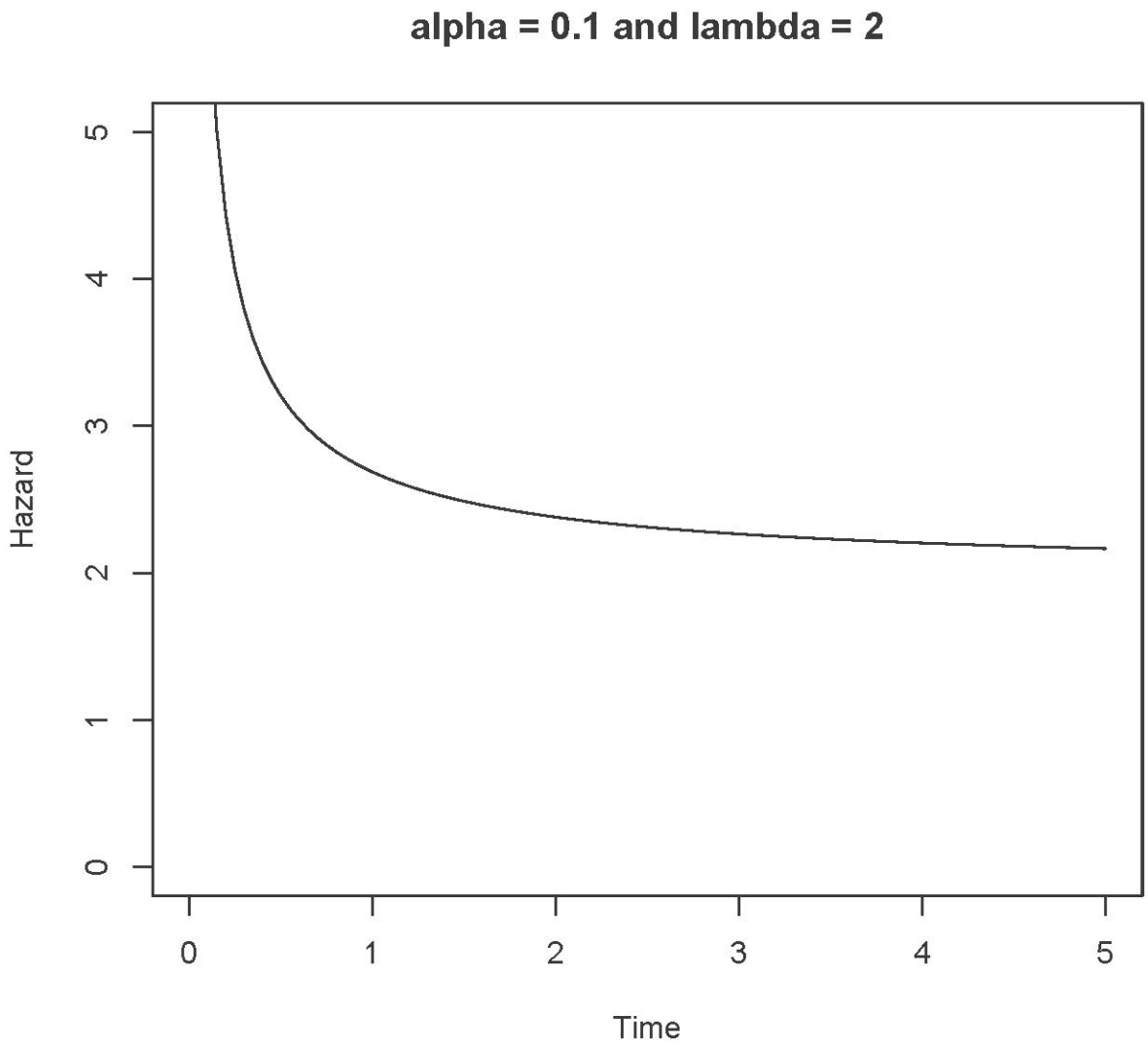



```
> # So far, we know that
>     # If alpha=1, h(t) is constant at lambda
>     # If alpha > 1, h(t) is increasing and bounded above by lambda
>     # If alpha < 1, h(t) is decreasing and bounded below by lambda
>     # alpha values close to one may make the big change occur closer to zero.
```

```
> # Now alpha < 1 again
> alpha=1/2;lambda=2
> Time = seq(from=0,to=5,length=101)
> logh = dgamma(Time,shape=alpha,rate=lambda,log=TRUE) -
+     pgamma(Time,shape=alpha,rate=lambda, lower.tail=FALSE,log.p=TRUE)
> Hazard = exp(logh)
> tstring = paste('alpha =',alpha,'and lambda =',lambda)
> plot(Time,Hazard,type='l',main=tstring,ylim=c(0,5)) # That's a lower case ell
```



```
> alpha=1/10;lambda=2
> Time = seq(from=0,to=5,length=101)
> logh = dgamma(Time,shape=alpha,rate=lambda,log=TRUE) -
+     pgamma(Time,shape=alpha,rate=lambda, lower.tail=FALSE,log.p=TRUE)
> Hazard = exp(logh)
> tstring = paste('alpha =',alpha,'and lambda =',lambda)
> plot(Time,Hazard,type='l',main=tstring,ylim=c(0,5)) # That's a lower case ell
```



```
> # Is the hazard function asymptotic to lambda?
>
> alpha=1/10;lambda=2
> Time = seq(from=0,to=50,length=101)
> logh = dgamma(Time,shape=alpha,rate=lambda,log=TRUE) -
+     pgamma(Time,shape=alpha,rate=lambda, lower.tail=FALSE,log.p=TRUE)
> Hazard = exp(logh)
> tstring = paste('alpha =',alpha,'and lambda =',lambda)
> plot(Time,Hazard,type='l',main=tstring,ylim=c(0,5)) # That's a lower case ell
>
> # Add the line y = lambda = 2
> xx = c(0,50); yy =c(2,2); lines(xx,yy,lty=2)
```